Evaluating Differentially Private Stochastic Gradient Descent Optimizers for Training Transformer Language Models

Eric Gong, Oleg Pavliv, Max Peng

Harvard John A. Paulson School of Engineering and Applied Sciences

HARVARD UNIVERSITY

Background and Problem Statement

The remarkable ability of LLMs to generalize to a diverse variety of tasks is made possible by training upon immense corpuses of data. These data, scraped from internet sources with little to no filtering and oversight can risk the contamination of LLM's model weights with sensitive and private information, which may now become readily available to all parties utilizing the model. As such, providing formal guarantees on limiting the extent to which any particular secret or private information found in the training data can influence model weights is of critical interest.

We implement transformer language models with differentially private stochastic gradient descent optimizers, evaluating trade-offs between privacy guarantees and model performance

Differential Privacy in Model Training

Algorithm 1 Differentially Private Stochastic Gradient Descent [1]

```
for $t = 1$ to $T$ do
    Sample $B_t$ of size $b$.
    for each $x \in B_t$ do
        $\bar{g}_t(x) \leftarrow \nabla_\theta L(\theta_{t-1}, x) / \max\big(1, \|g_t(x)\|_2 / C\big)$.
    end for
    $\widetilde{g}_t \leftarrow \frac{1}{b} \sum_{x \in B_t} \bar{g}_t(x) + \mathcal{N}(0, \sigma^2 C^2 I)$.
    $\theta_t \leftarrow \theta_{t-1} - \eta_t \, \widetilde{g}_t$.
end for
```

1 Naive Composition Theorem

We assume that each iteration of the optimizer is an (ε', δ')-DP release. For an overall (ε, δ)-DP privacy release, we require gaussian noise defined by the following:

$$\sigma_{\text{naive}} = \frac{\sqrt{2 \ln(1.25/\delta')}}{\varepsilon'} = \frac{q\,T\,\sqrt{2 \ln\!\big(\frac{1.25\,q\,T}{\delta}\big)}}{\varepsilon}.$$

2 Advanced Composition Theorem

Under Advanced Composition [2], the composition across T (ε', δ')-DP iterations yields an overall $(\widetilde{\varepsilon},\ T\delta' + \delta'')$-DP guarantee, which simplifies to the following:

$$\widetilde{\varepsilon} = \varepsilon' \sqrt{2T \ln \tfrac{1}{\delta''}} + T\varepsilon'(e^{\varepsilon'} - 1) \implies \varepsilon < \widetilde{\varepsilon} \approx \frac{\varepsilon'}{2\sqrt{2T\ \ln\!\big(1/\delta'\big)}}.$$

Advanced Composition accounts for shared data across each iteration. Thus, it suffices to have gaussian noise addition defined by the following equation:

$$\sigma_{\text{advanced}} = O\Big(\frac{q\,\sqrt{T\ \ln(1/\delta)\ \ln(T/\delta)}}{\varepsilon}\Big).$$

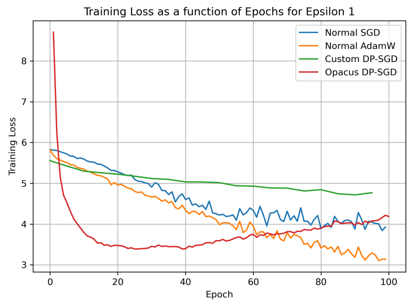
3 DP-SGD Specific Composition methods

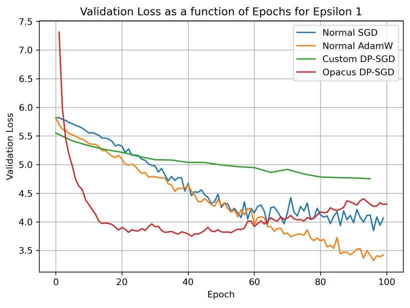
Specific properties of SGD optimizers enable DP-SGD specific bounds on Gaussian noise. The Moments Accountant method [1] establishes that for a Gaussian mechanism to preserve (ε, δ)-DP under T-fold composition, it suffices to have:

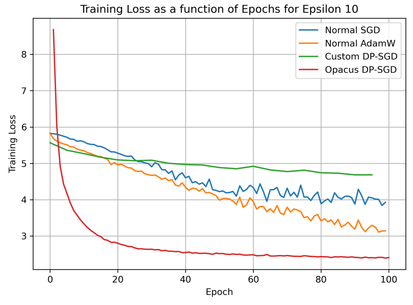
$$\sigma_{\text{moments}} \geq O(\frac{q\,\sqrt{T\ \ln(1/\delta)}}{\varepsilon})$$

Commercial implementations—such as Opacus—scale noise as a function of epoch counts [3], reducing utility loss while maintaining differential privacy guarantees.

Model Loss Trends Across Varying Epsilon







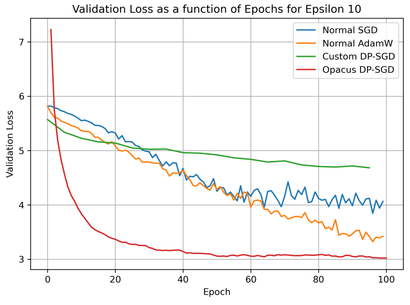


Table 1: Ranking Models on Low-Epoch Training Runs (Best = 1)

Privacy Budget (ε)	SGD[†]	AdamW[†]	Custom-DP	Opacus
$\varepsilon = 1$	2	1	4	3
$\varepsilon = 10$	3	2	4	1

[†] Baseline models with non-DP optimizers unaffected by varying epsilon

Methodology

We implement a simple transformer model complete with multiple self-attention heads, layer normalization, residual connections, random dropout, etc. A simple word-level tokenizer allows for fine-grained control of the vocabulary size, scaling experiments within compute confines.

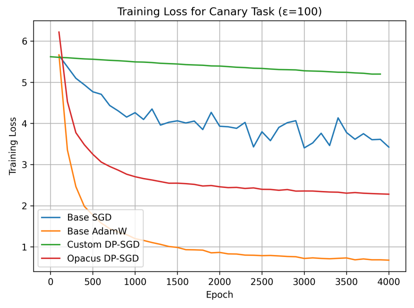
We create a Custom DP-SGD model by integrating the base transformer model with a DP-SGD optimizer implemented by overwriting the PyTorch SGD class. This serves as a direct comparison between the base SGD model and a model utilizing the moments accounting method to provide DP guarantees.

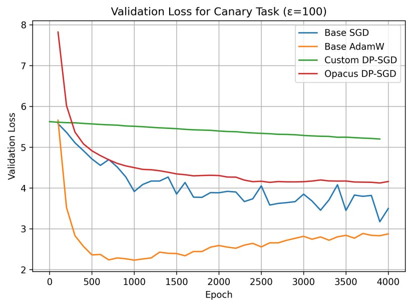
In addition we implement a second DP-SGD model in Opacus, given its ease of integration with the PyTorch library utilized in the base transformer model. Opacus employs a PrivacyEngine abstraction to track the privacy budget and operate on model gradients by attaching to a standard PyTorch optimizer.

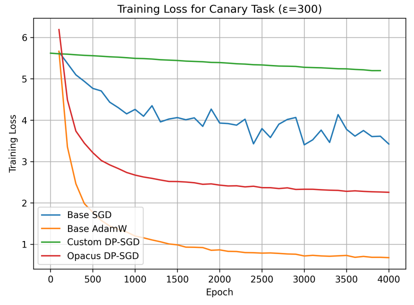
We compare two variations of non-DP implementations with a SGD and AdamW optimizer to our Custom and Opacus DP variants under two experimental situations. In the first, we train the DP models for a small number of epochs, examining trends between differing models under differing degrees of privacy guarantees.

In the second experimental situation, we insert strategic secrets—known as canaries—into the training text, and evaluate whether each model exposes the secret canary after an extended number of training epochs.

Evaluation Against Training Data Canaries







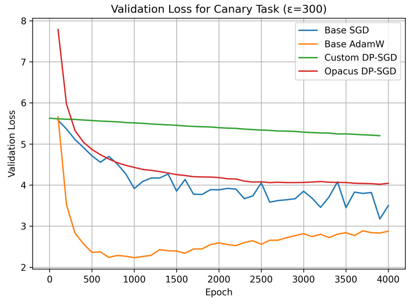


Table 2: Canary Disclosure Across Varying DP and Non-DP Models

Privacy Budget (ε)	SGD[†]	AdamW[†]	Custom-DP	Opacus
$\varepsilon = 100$	Safe	Disclosed	Safe	Safe
$\varepsilon = 300$	Safe	Disclosed	Safe	Safe

[†] Baseline models with non-DP optimizers unaffected by varying epsilon

Conclusions

DP SGD offers privacy guarantees at the cost of performance on large training runs: non-DP variants (such as AdamW optimizers) asymptotically outperform the training loss and validation loss of even professional DP SGD models.

However, the guarantees provided by DP SGD exhibit beneficial outcomes in terms of model performance. On training runs for which the number of epochs is low, DP SGD models results in faster convergence, with the addition of gaussian noise preventing the model from falling into local minima.

In addition, for large epochs, the privacy guarantees of DP-SGD subverts model overfitting, reducing performance degradation during validation. As such, we find DP SGD to be a useful tool, that although not without drawbacks, may have potential use in specific data analysis and training situations.

References

[1] M. Abadi, A. Chu, I. Goodfellow, B. McMahan, I. Mironov, K. Talwar, and L. Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 308–318. ACM, 2016.

[2] C. Dwork and A. Roth. *The Algorithmic Foundations of Differential Privacy*, volume 9 of *Foundations and Trends in Theoretical Computer Science*. Now Publishers, 2014. ISBN 978-1-60845-350-1.

[3] Meta Platforms, Inc. Privacy engine. Opacus API Documentation, 2025. `https://opacus.ai/api/privacy_engine.html`.